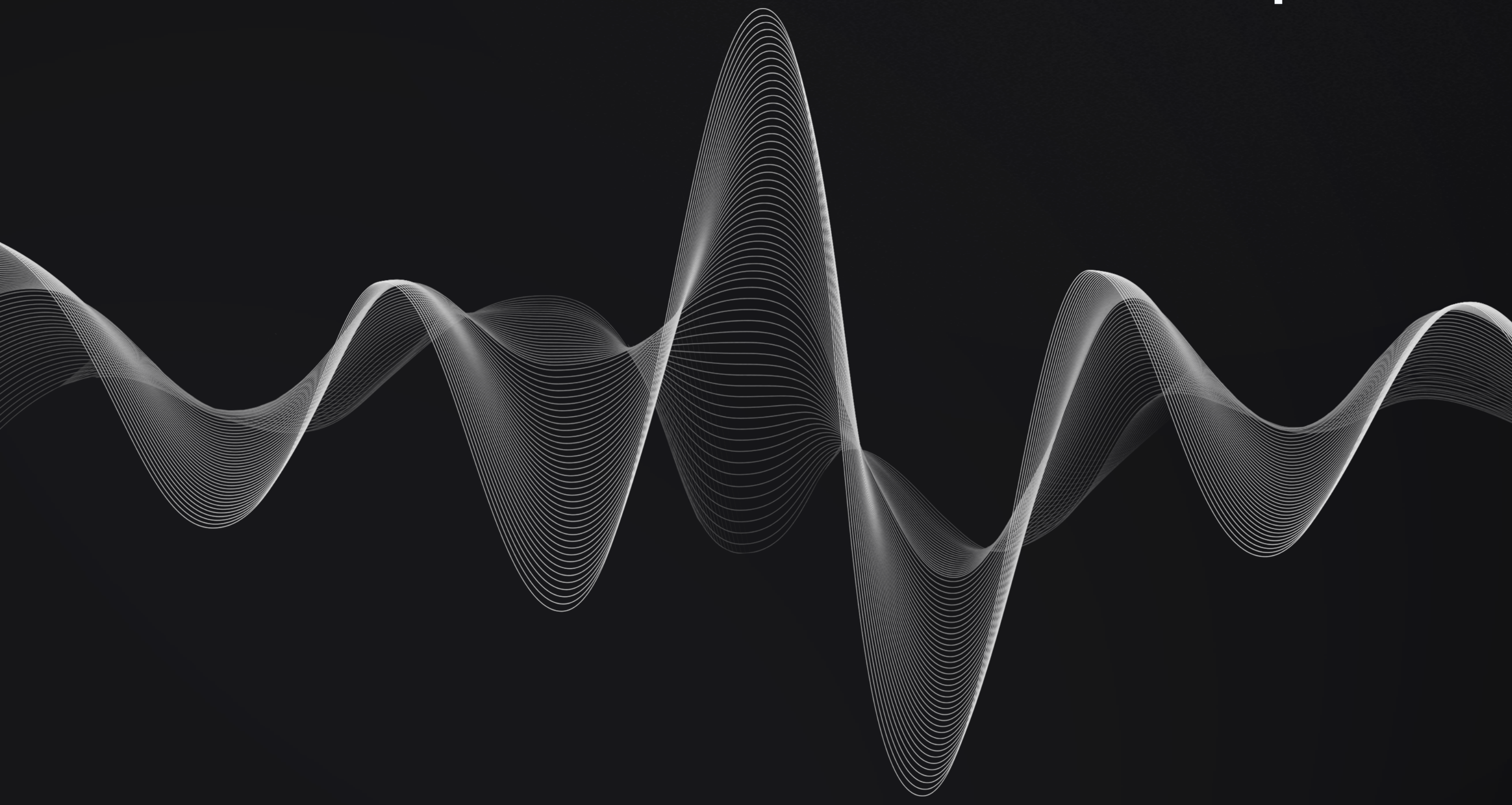
RESONANCE

deploi

Deploi

Deploi Notes Smart Contract Audit Report



Blockchain, Emerging Technology, and Web2
CYBERSECURITY PRODUCT & SERVICE ADVISORY

Document Control

PUBLIC **FINAL**(v2.1)

Audit_Report_DEPL-NOT_FINAL_21

Oct 17, 2025	○	v0.1	Luis Arroyo: Initial draft
Oct 17, 2025	○	v0.2	Luis Arroyo: Added findings
Oct 17, 2025	🟢	v1.0	Charles Dray: Approved
Oct 22, 2025	○	v1.1	Luis Arroyo: Reviewed findings
Oct 22, 2025	🟡	v2.0	Charles Dray: Finalized
Oct 23, 2025	○	v2.1	Charles Dray: Published

Points of Contact	Oskars Jepsis	Deploi	oskars.j@deploi.org
	Charles Dray	Resonance	charles@resonance.security
Testing Team	Luis Arroyo	Resonance	luis.arroyo@resonance.security
	João Simões	Resonance	joao@resonance.security
	Ilan Abitbol	Resonance	ilan@resonance.security

Copyright and Disclaimer

The information in this report is considered confidential and proprietary by Resonance and is licensed to the recipient solely under the terms of the project statement of work. Reproduction or distribution, in whole or in part, is strictly prohibited without the express written permission of Resonance.

All activities performed by Resonance in connection with this project were carried out in accordance with the project statement of work and agreed-upon project plan. It's important to note that security assessments are time-limited and may depend on information provided by the client, its affiliates, or partners. As such, the findings documented in this report should not be considered a comprehensive list of all security issues, flaws, or defects in the target system or codebase.

Furthermore, it is hereby assumed that all of the risks in electing not to remedy the security issues identified henceforth are sole responsibility of the respective client. The acknowledgement and understanding of the risks which may arise due to failure to remedy the described security issues, waives and releases any claims against Resonance, now known or hereafter known, on account of damage or financial loss.

Contents

1 **Document Control** 2

Copyright and Disclaimer 2

2 **Executive Summary** 4

System Overview 4

Repository Coverage and Quality 4

3 **Target** 6

4 **Methodology** 7

Severity Rating 8

Repository Coverage and Quality Rating 9

5 **Findings** 10

Unable To Obtain ERC-3475 classValues/nonceValues 11

No Class Or Nonce Validation When Calling issue() 12

Whitelist Bypass For Delegates 13

A **Proof of Concepts** 14

Executive Summary

Deploi contracted the services of Resonance to conduct a comprehensive security audit of their smart contracts between October 15, 2025 and October 17, 2025. The primary objective of the assessment was to identify any potential security vulnerabilities and ensure the correct functioning of smart contract operations.

During the engagement, Resonance allocated 2 engineers to perform the security review. The engineers, including an accomplished professional with extensive proficiency in blockchain and smart-contract security, encompassing specialized skills in advanced penetration testing, and in-depth knowledge of multiple blockchain protocols, devoted 3 days to the project. The project's test targets, overview, and coverage details are available throughout the next sections of the report.

The ultimate goal of the audit was to provide Deploi with a detailed summary of the findings, including any identified vulnerabilities, and recommendations to mitigate any discovered risks. The results of the audit are presented in detail further below.

Conclusion

The audit outcomes affirm the project's strong commitment to security and quality. With all identified issues resolved, the final review indicates that the smart contracts are well-engineered, resilient, and aligned with industry-leading security standards.

System Overview

DeploiNotes3475 is an upgradeable ERC-3475 implementation that issues "Deploi Private Credit Notes" across class/nonce tranches.

Governance hinges on role-based access where default admin curates metadata and participant status while operators conduct issuance, redemption, and burn of the whitelisted user notes.

Repository Coverage and Quality

Code	Tests	Documentation
9 / 10	10 / 10	9 / 10

Resonance's testing team has assessed the Code, Tests, and Documentation coverage and quality of the system and achieved the following results:

- The code follows development best practices and makes use of known patterns, standard libraries, and language guides. It is easily readable and uses the latest stable version of relevant components. Overall, **code quality is excellent**.

- Unit and integration tests are included. The tests cover both technical and functional requirements. Overall, **tests coverage and quality is excellent**.
    - Function Coverage: 100% (33/33 functions)
    - Branch Coverage: 87% (104/120 branches)
    - Line Coverage: 91% (542/595 lines)
- The documentation includes the specification of the system, technical details for the code, relevant explanations of workflows and interactions. Overall, **documentation coverage and quality is excellent**.

Target

The objective of this project is to conduct a comprehensive review and security analysis of the smart contracts that are contained within the specified repository.

The following items are included as targets of the security assessment:

- Repository: `deploi-assets/deploi-erc3475/`
- Hash: 13847fdd421205a6cc7eae9d6922baa43af844dc

The following items are excluded:

- External and standard libraries
- Files pertaining to the deployment process
- Financial related attacks

Methodology

In the context of security audits, Resonance's primary objective is to portray the workflow of a real-world cyber attack against an entity or organization, and document in a report the findings, vulnerabilities, and techniques used by malicious actors. While several approaches can be taken into consideration during the assessment, Resonance's core value comes from the ability to correlate automated and manual analysis of system components and reach a comprehensive understanding and awareness with the customer on security-related issues.

Resonance implements several and extensive verifications based off industry's standards, such as, identification and exploitation of security vulnerabilities both public and proprietary, static and dynamic testing of relevant workflows, adherence and knowledge of security best practices, assurance of system specifications and requirements, and more. Resonance's approach is therefore consistent, credible and essential, for customers to maintain a low degree of risk exposure.

Ultimately, product owners are able to analyze the audit from the perspective of a malicious actor and distinguish where, how, and why security gaps exist in their assets, and mitigate them in a timely fashion.

Source Code Review - Solidity EVM

During source code reviews for Web3 assets, Resonance includes a specific methodology that better attempts to effectively test the system in check:

1. Review specifications, documentation, and functionalities
2. Assert functionalities work as intended and specified
3. Deploy system in test environment and execute deployment processes and tests
4. Perform automated code review with public and proprietary tools
5. Perform manual code review with several experienced engineers
6. Attempt to discover and exploit security-related findings
7. Examine code quality and adherence to development and security best practices
8. Specify concise recommendations and action items
9. Revise mitigating efforts and validate the security of the system

Additionally and specifically for Solidity EVM audits, the following attack scenarios and tests are recreated by Resonance to guarantee the most thorough coverage of the codebase:

- Reentrancy attacks
- Frontrunning attacks
- Unsafe external calls
- Unsafe third party integrations
- Denial of service
- Access control issues

- Inaccurate business logic implementations
- Incorrect gas usage
- Arithmetic issues
- Unsafe callbacks
- Timestamp dependence
- Mishandled panics, errors and exceptions

Severity Rating

Security findings identified by Resonance are rated based on a Severity Rating which is, in turn, calculated off the **impact** and **likelihood** of a related security incident taking place. This rating provides a way to capture the principal characteristics of a finding in these two categories and produce a score reflecting its severity. The score can then be translated into a qualitative representation to help customers properly assess and prioritize their vulnerability management processes.

The **impact** of a finding can be categorized in the following levels:

1. Weak **-** Inconsequential or minimal damage or loss
2. Medium **-** Temporary or partial damage or loss
3. Strong **-** Significant or unrecoverable damage or loss

The **likelihood** of a finding can be categorized in the following levels:

1. Unlikely **-** Requires substantial knowledge or effort or uncontrollable conditions
2. Likely **-** Requires technical knowledge or no special conditions
3. Very Likely **-** Requires trivial knowledge or effort or no conditions

			Likelihood	
		Very Likely	Likely	Unlikely
Impact	Strong	Critical	High	Medium
	Medium	High	Medium	Low
	Weak	Medium	Low	Info

Repository Coverage and Quality Rating

The assessment of Code, Tests, and Documentation coverage and quality is one of many goals of Resonance to maintain a high-level of accountability and excellence in building the Web3 industry. In Resonance it is believed to be paramount that builders start off with a good supporting base, not only development-wise, but also with the different security aspects in mind. A product, well thought out and built right from the start, is inherently a more secure product, and has the potential to be a game-changer for Web3's new generation of blockchains, smart contracts, and dApps.

Accordingly, Resonance implements the evaluation of the code, the tests, and the documentation on a score **from 1 to 10** (1 being the lowest and 10 being the highest) to assess their quality and coverage. In more detail:

- Code should follow development best practices, including usage of known patterns, standard libraries, and language guides. It should be easily readable throughout its structure, completed with relevant comments, and make use of the latest stable version components, which most of the times are naturally more secure.
- Tests should always be included to assess both technical and functional requirements of the system. Unit testing alone does not provide sufficient knowledge about the correct functioning of the code. Integration tests are often where most security issues are found, and should always be included. Furthermore, the tests should cover the entirety of the codebase, making sure no line of code is left unchecked.
- Documentation should provide sufficient knowledge for the users of the system. It is useful for developers and power-users to understand the technical and specification details behind each section of the code, as well as, regular users who need to discern the different functional workflows to interact with the system.

Findings

During the security audit, several areas for improvement were identified and subsequently addressed, resulting in a stronger and more robust codebase. These findings, represented by unique IDs, are detailed in this section with relevant information including Severity, Category, Status, Code Section, Description, and Recommendation. Further extensive information may be included in corresponding appendices should it be required.

An overview of all the identified findings is outlined in the table below, where they are sorted by Severity and include a **Remediation Priority** metric asserted by Resonance's Testing Team. This metric characterizes findings as follows:

- ııııııı **"Quick Win"** Requires little work for a high impact on risk reduction.
- ılıllı **"Standard Fix"** Requires an average amount of work to fully reduce the risk.
- ı|ı|||ı **"Heavy Project"** Requires extensive work for a low impact on risk reduction.

RES-01	🟧	Unable To Obtain ERC-3475 classValues/nonceValues	ılıllı	Resolved
RES-02	🟩	No Class Or Nonce Validation When Calling issue()	ılıllı	Resolved
RES-03	🟩	Whitelist Bypass For Delegates	ılıllı	Resolved

Unable To Obtain ERC-3475 classValues/nonceValues

Medium **RES-DEPL-NOT01** Data Validation **Resolved**

Code Section

- `contracts/DeploiNotes3475.sol#L241`
- `contracts/DeploiNotes3475.sol#L246`

Description

Neither `_classes[c].valuesId[id]` nor `_classes[c].nonces[n]._valuesId[id]` are ever populated, so calls to `classValues` and `nonceValues` always revert with "`VAL`".

Integrators relying on the standard interface cannot read value metadata, breaking compliance and any off-chain logic that depends on it.

Recommendation

It is recommended to populate the index-to-key mappings or expose direct string-based getters.

Status

The issue has been fixed in 5939155967c3ab70891f6ffb9a20eadeda6d7a8e.

No Class Or Nonce Validation When Calling issue()

Low RES-DEPL-NOT02 Data Validation **Resolved**

Code Section

- `contracts/DeploiNotes3475.sol#L167`

Description

The `issue()` function trusts the caller to pick existing issuances. Supplying a fresh classId/nonceId implicitly creates it with zero metadata/terms. This may allow minting "ghost" instruments or bypass the issuer intended functionality.

Recommendation

It is recommended to verify that metadata exists or revert if a new supply is detected.

Status

The issue has been fixed in 5939155967c3ab70891f6ffb9a20eadeda6d7a8e.

Whitelist Bypass For Delegates

Low

RES-DEPL-NOT03 Data Validation **Resolved**

Code Section

- contracts/DeploiNotes3475.sol#L96
- contracts/DeploiNotes3475.sol#L101

Description

`setApprovalFor` and `approve` let holders authorize any address, but `transferFrom` and `transferAllowanceFrom` only ensure `from` and `to` are on the whitelist, allowing a non-whitelisted account to receive approval and move funds between whitelisted accounts.

Recommendation

It is recommended that delegate addresses (operator or allowance spender) are whitelisted when the approval is granted or when they execute.

Status

The issue has been fixed in 5939155967c3ab70891f6ffb9a20eadeda6d7a8e.

Proof of Concepts

No Proof-of-Concept was deemed relevant to describe findings in this engagement.